AFRL-AFOSR-JP-TR-2016-0046

Designing Feature and Data Parallel Stochastic Coordinate Descent Method for Matrix and Tensor Factorization

U Kang
Korea Advanced Institute of Science and Technology

05/11/2016
Final Report

Air Force Research Laboratory
AF Office Of Scientific Research (AFOSR)/ IOA
Arlington, Virginia 22203
Air Force Materiel Command

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
11-05-2016	Final	30 Apr 2014 to 29 Apr 2016

4. TITLE AND SUBTITLE
Designing Feature and Data Parallel Stochastic Coordinate Descent Method for Matrix and Tensor Factorization

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA2386-14-1-4036

5c. PROGRAM ELEMENT NUMBER
61102F

6. AUTHOR(S)
U Kang

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Korea Advanced Institute of Science and Technology
291 Daehak-ro, Yuseong-gu
Taejon, 305701 KR

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
AOARD
UNIT 45002
APO AP 96338-5002

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/AFOSR IOA

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-AFOSR-JP-TR-2016-0046

12. DISTRIBUTION/AVAILABILITY STATEMENT
A DISTRIBUTION UNLIMITED: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Given a high-order large-scale tensor, how can we decompose it into latent factors? Can we process it on commodity computers with limited memory? These questions are closely related to recommender systems, which have modeled rating data not as a matrix but as a tensor to utilize contextual information such as time and location. This increase in the order requires tensor factorization methods scalable with both the order and size of a tensor. We proposed two distributed tensor factorization methods, CDTF and SALS. Both methods are scalable with all aspects of data and show a trade-off between convergence speed and memory requirements. CDTF, based on coordinate descent, updates one parameter at a time, while SALS generalizes on the number of parameters updated at a time. In our experiments, only our methods factorized a 5-order tensor with 1 billion observable elements, 10M mode length, and 1K rank, while all other state-of-the-art methods failed. Moreover, our methods required several orders of magnitude less memory than our competitors. We implemented our methods on MAPREDUCE with two widely-applicable optimization techniques: local disk caching and greedy row assignment. They speeded up our methods up to 98.2x and also the competitors up to 5.9x.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			LUTZ, BRIAN
Unclassified	Unclassified	Unclassified	SAR		

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

				19b. TELEPHONE NUMBER *(Include area code)* 315-227-7006

Final Report for AOARD Grant FA2386-14-1-4036

"Designing Feature and Data Parallel Stochastic Coordinate Descent Method for Matrix and Tensor Factorization"

29 April, 2016

Name of Principal Investigators (PI and Co-PIs): U kang

- e-mail address : ukang@cs.kaist.ac.kr
- Institution : Department of Computer Science, Korea Advanced Institute of Science and Technology
- Mailing Address : 291 Daehak-ro, Yuseong-gu, Daejeon, Republic of Korea
- Phone : 82-42-350-3565
- Fax : 82-42-350-7765

Period of Performance: April/30/2014 – April/29/2016

Abstract:

Given a high-order large-scale tensor, how can we decompose it into latent factors? Can we process it on commodity computers with limited memory? These questions are closely related to recommender systems, which have modeled rating data not as a matrix but as a tensor to utilize contextual information such as time and location. This increase in the order requires tensor factorization methods scalable with both the order and size of a tensor. In this paper, we propose two distributed tensor factorization methods, CDTF and SALS. Both methods are scalable with all aspects of data and show a trade-off between convergence speed and memory requirements. CDTF, based on coordinate descent, updates one parameter at a time, while SALS generalizes on the number of parameters updated at a time. In our experiments, only our methods factorized a 5-order tensor with 1 billion observable elements, 10M mode length, and 1K rank, while all other state-of-the-art methods failed. Moreover, our methods required several orders of magnitude less memory than our competitors. We implemented our methods on MAPREDUCE with two widely-applicable optimization techniques: local disk caching and greedy row assignment. They speeded up our methods up to 98.2x and also the competitors up to 5.9x.

Introduction:

The recommendation problem can be viewed as completing a partially observable user-item matrix whose entries are ratings. Matrix factorization (MF), which decomposes the input matrix into a user factor matrix and an item factor matrix so that their multiplication approximates the input matrix, is one of the most widely used methods. On the other hand, there have been attempts to improve the accuracy of recommendation by using additional information such as time and location. A straightforward way to utilize such extra factors is to model rating data as a partially observable tensor where additional dimensions correspond to the extra factors. As in the matrix case, tensor factorization (TF), which decomposes the input tensor into multiple factor matrices and a core tensor, has been used. As the dimension of web-scale recommendation problems increases, a necessity for TF algorithms scalable with the dimension as well as the size of data has arisen.

The goal of the whole project is to investigate feature and data parallel stochastic coordinate descent method for matrix and tensor factorization, which has three advantages: 1) fully distributed, and can solve matrix and tensor factorization problems quickly using peta-scale data, 2) has theoretical converges guarantee, and 3) makes impact to broad applications, including recommendation and trend analysis, since any matrix or tensor data can be handled.

Experiment:
The experiments are performed to answer the following questions.
- What is the data scalability of the proposed method?
- What is the machine scalability of the proposed method?
- How fast does the proposed method converge?

We compare our proposed methods CDTF and SALS, with competitors including PSGD, ALS, and FlexiFact. The experiments are run on a 40-node Hadoop cluster. Each node has an Intel Xeon E5620 2.4GHz CPU. The maximum heap memory size per reducer is set to 8GB. We use both the real-world tensors and the synthetic tensors. All the methods are implemented using Java with Hadoop 1.0.3.
We use both real-world and synthetic datasets which are listed in Tables 1 and 2.

Table 1: Scale of Synthetic Datasets. N: dimension, I: length of a mode, |Ω|: # of nonzeros, K: rank.

	S1	**S2** (default)	**S3**	**S4**
N	2	3	4	5
I	300K	1M	3M	10M
$\|\Omega\|$	30M	100M	300M	1B
K	30	100	300	1K

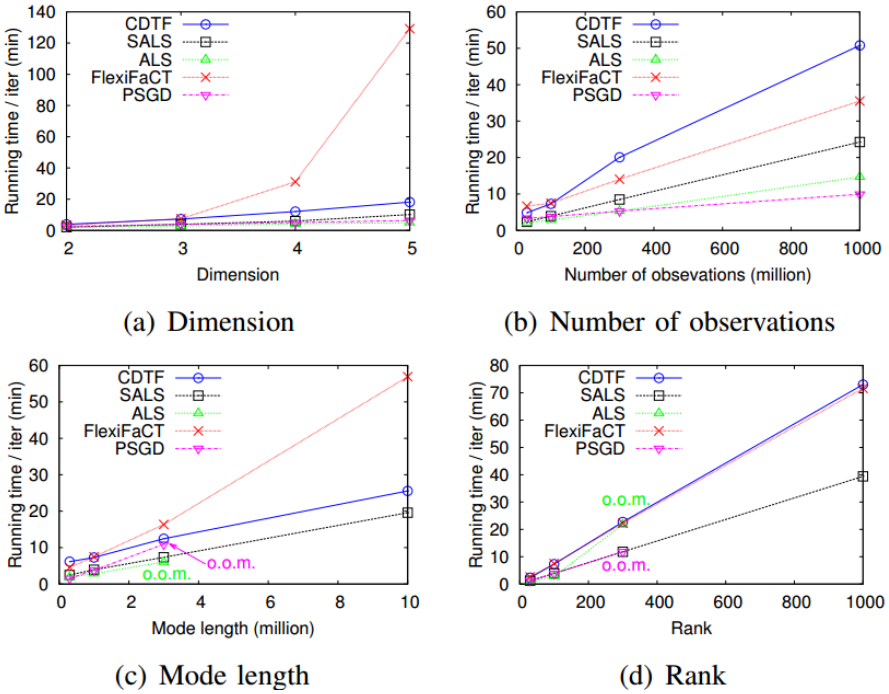
Table 2: Scale of real world datasets. N: dimension, I_1~I_4: length of a mode, |Ω|: # of nonzeros in training data, $|\Omega|_{test}$: # of nonzeros in test data, K: rank, λ: regularization parameter, η_0: initial running rate.

	Movielens$_4$	**Netflix**$_3$	**Yahoo-music**$_4$
N	4	3	4
I_1	71,567	2,649,429	1,000,990
I_2	65,133	17,770	624,961
I_3	169	74	133
I_4	24	-	24
$\|\Omega\|$	9,301,274	99,072,112	252,800,275
$\|\Omega\|_{test}$	698,780	1,408,395	4,003,960
K	20	40	80
λ	0.01	0.02	1.0
η_0	0.01	0.01	10^{-5} (FLEXIFACT) 10^{-4} (PSGD)

Results and Discussion:

1) Results and Discussion

- Data Scalability
We measure the scalability of CDTF, SALS, and the competitors with regard to the dimension, number of observations, mode length, and rank of an input tensor. When measuring the scalability with regard to a factor, the factor is scaled up from S1 to S4 while all other factors are fixed at S2 as summarized in Table 1. As seen in Figure 1(a), FLEXIFACT does not scale with dimension because of its communication cost, which increases exponentially with dimension. ALS and PSGD are not scalable with mode length and rank due to their high memory requirements as Figures 1(c) and 1(d) show. They require up to 11.2GB, which is 48× of 234MB that CDTF requires and 10× of 1,147MB that SALS requires. Moreover, the running time of ALS increases rapidly with rank owing to its cubically increasing computational cost. Only SALS and CDTF are scalable with all the factors. Their running times increase linearly with all the factors except dimension, with which they increase slightly faster due to the quadratically increasing computational cost.



(a) Dimension (b) Number of observations

(c) Mode length (d) Rank

Figure1. Data scalability.

- Machine Scalability

We measure the speed-ups (T_5/T_M where T_M is the running time with M reducers) of the methods on the S2 scale dataset by increasing the number of reducers. The result is shown in Figure 2. The speed-ups of CDTF, SALS, and ALS increase linearly at the beginning and then flatten out slowly owing to their fixed communication cost which does not depend on the number of reducers. The speed-up of PSGD flattens out fast, and PSGD even slightly slows down in 40 reducers because of increased overhead. FLEXIFACT slows down as the number of reducers increases because of its rapidly increasing communication cost.

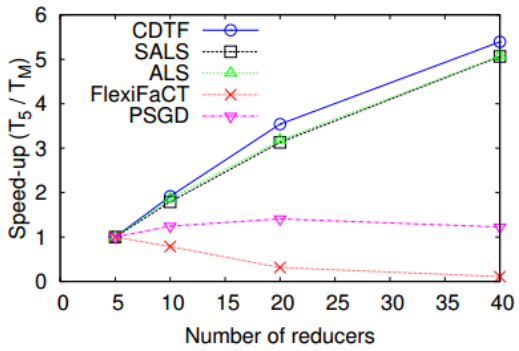


Figure 2. Machine scalability.

- Convergence

We compare how quickly and accurately each method factorizes real-world tensors using the following models.

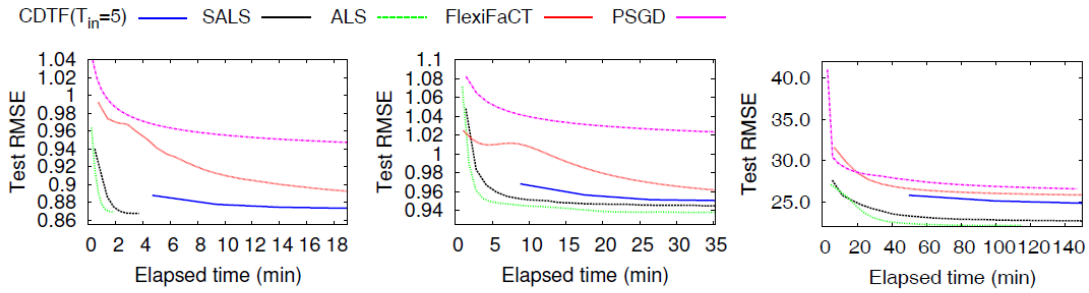
- PARAFAC model (Figure 3).
- Bias model (Figure 4).
- L2 regularization model (Figure 5).
- L2 regularization with non-negativity model (Figure 6).
- L1 regularization (Figure 7).

- Coupled tensor factorization (Figure 8).

Accuracies are calculated at each iteration by root mean square error (RMSE) on a held-out test set, which is a measure commonly used by recommendation systems. SALS and ALS are not shown in Figures 6 and 7 since they are not applicable to non-negativity constraint and L1 regularization.

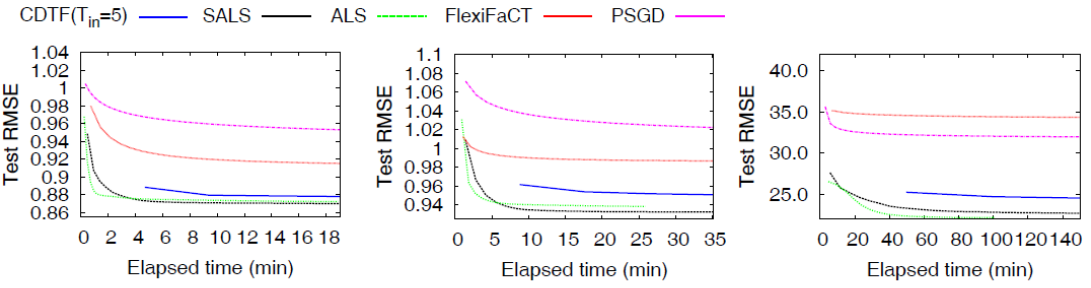
For PARAFAC, bias, L2 regularization, and coupled tensor factorization models, SALS is comparable with ALS, which converges the fastest to the best solution, and CDTF follows them. PSGD converges the slowest to the worst solution due to the non-identifiability of the optimization problem.

For non-negativity constraint and L1 regularization models, CDTF shows the best performance in terms of error and running time.



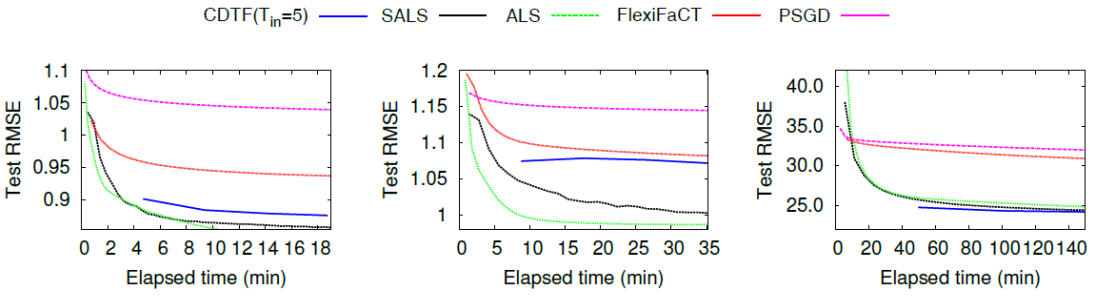
(a) Movielens$_4$ × 10 (PARAFAC) (b) Netflix$_3$ (PARAFAC) (c) Yahoo-music$_4$ (PARAFAC)

Figure 3. Convergence results of the PARAFAC model.



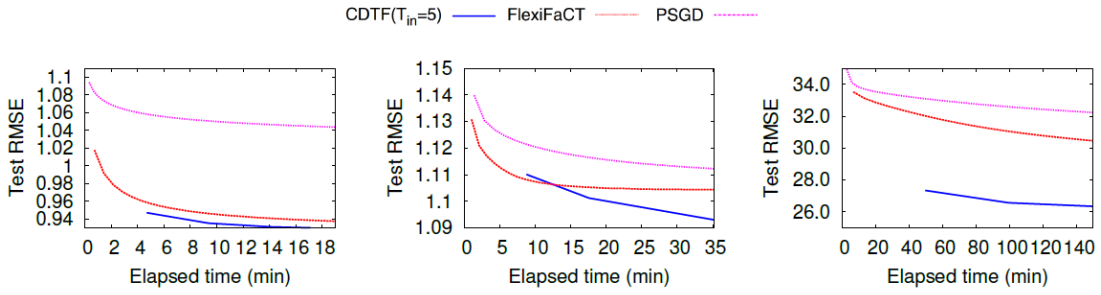
(a) Movielens$_4$ × 10 (Bias) (b) Netflix$_3$ (Bias) (c) Yahoo-music$_4$ (Bias)

Figure 4. Convergence results of the bias model.



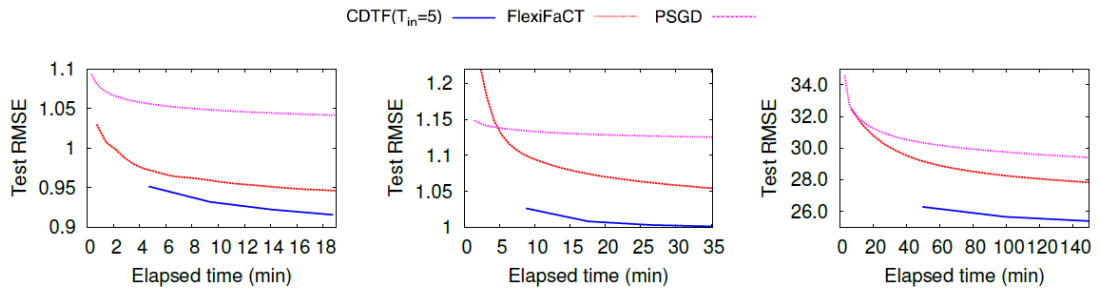
(a) Movielens$_4$ × 10 (b) Netflix$_3$ (c) Yahoo-music$_4$

Figure 5. Convergence results of the L2 regularization model.



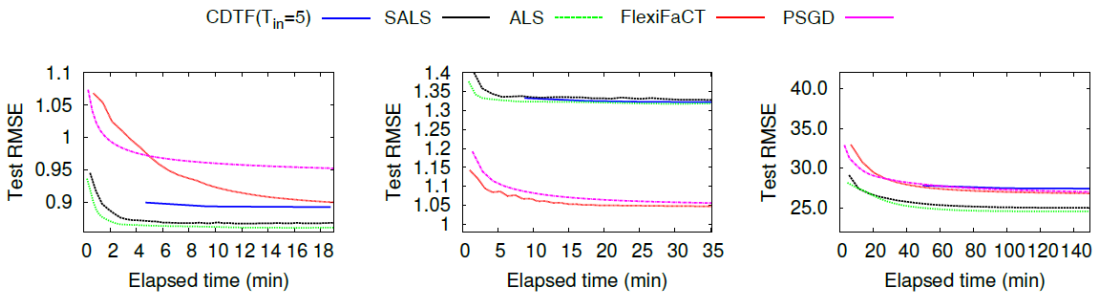
(a) Movielens$_4$ × 10 (b) Netflix$_3$ (c) Yahoo-music$_4$

Figure 6. Convergence results of the L2 regularization with non-negativity model.



(a) Movielens$_4$ × 10 (b) Netflix$_3$ (c) Yahoo-music$_4$

Figure 7. Convergence results of the L1 regularization model.



(a) Movielens$_4$ × 10 (b) Netflix$_3$ (c) Yahoo-music$_4$

Figure 8. Convergence results of the coupled tensor factorization model.

- Discussion

The most famous algorithm for tensor factorization is the ALS (alternating least square). We showed that the proposed CDTF and SALS method provides better scalability, running time, and accuracy than ALS and other previous methods. CDTF and SALS can be used for very large scale tensor factorization where running time and scalability are of crucial importance. Future works include applying other optimization methods, including second-order methods, for scalable and distributed tensor factorization.

List of Publications and Significant Collaborations that resulted from your AOARD supported project: In standard format showing authors, title, journal, issue, pages, and date, for each category list the following:

a) papers published in peer-reviewed journals,

b) papers published in peer-reviewed conference proceedings,

- K. Shin and U. Kang, “Distributed methods for high dimensional and large-scale tensor factorization,” in 2014 IEEE International Conference on Data Mining, ICDM 2014, Shenzhen, China, December 14-17, 2014, pp. 989–994, 2014.

c) papers published in non-peer-reviewed journals and conference proceedings,
d) conference presentations without papers,
e) manuscripts submitted but not yet published, and
- K. Shin, Lee Sael, and U. Kang, “Fully Scalable Methods for Distributed Tensor Factorization,” submitted to IEEE Transactions on Knowledge and Data Engineering.

f) provide a list any interactions with industry or with Air Force Research Laboratory scientists or significant collaborations that resulted from this work.

Attachments: Publications a), b) and c) listed above if possible.